# Bluetooth Device Manager Connecting a Large Number of Resource-Constraint Devices in a Service-Oriented Bluetooth Network

Hendrik Bohn, Andreas Bobek, and Frank Golatowski

University of Rostock,
Institute for Applied Microelectronics and Computer Science,
Richard-Wagner-Strasse 31,
18119 Rostock-Warnemünde, Germany
{hendrik.bohn, andreas.bobek, frank.golatowski}@etechnik.uni-rostock.de

**Abstract.** The unique advantages of Bluetooth such as low power consumption capability, cheap hardware interfaces and easy set-up offer new application areas. This is a reason why Bluetooth is even considered for a Service Oriented Architectures (SOAs) consisting of a large number of resource-constraint devices. Although several proposals are available for reducing power consumption for intra-piconet communication, none of them addresses the utilization of the Park mode to reduce power consumption together with the support of a large number of accessible devices. This ongoing research work bridges that gap by proposing polling based on the probability of service access for a centralized Bluetooth network of resource-constraint devices. Services of connected devices (slaves) are offered to the central device (master) which manages all communications in the network. The slaves remain in Park mode unless their services are accessed by the master. This research work shows the feasibility of our proposal.

## 1 Introduction

Although Bluetooth [1] was originally designed as a cable replacement technology, its unique advances such as low power consumption, cheap hardware interfaces and easy set-up offer further application areas. Bluetooth can be used for a *Service Oriented Architecture* (SOA) which in our case connects a large number of resource-constraint devices (slaves) to a central device (master). SOAs are network architectures in which devices offer services to each other. *Services* are entities which provide information, perform an action or control resources on the behalf of other entities.

The application of Bluetooth in a service-oriented network of resource-constraint devices leads to two main problems. Firstly, a Bluetooth piconet supports only up to 8 devices in an active connection with a maximum bandwidth of 1 Mbps. Secondly, many services are only accessed once in a while and stay idle for most of the time although their Bluetooth interface are still in an active mode.

P. Lorenz and P. Dini (Eds.): ICN 2005, LNCS 3420, pp. 430–437, 2005.

The simplest configuration of a Bluetooth networks is a *piconet*, where up to 7 devices (slaves) are actively connected to a device (master) which manages all connections. Actively connected slaves are polled by the master and may send their data. The polling scheme in a piconet is not specified by the Bluetooth specification. Beside active connections the master can manage up to 255 slaves in *Park mode*. Parked slaves listen to the master in a certain interval and remain inactive on low power in between.

The probability of service access ranges from rarely (e.g. outside temperature service) to continuously (e.g. multimedia services). These different requirements for the connection provide new opportunities for power saving approaches. Devices may remain in low power mode unless their services are accessed. After service access associated devices switch to low power mode again.

This paper describes a device manager on a central device (master) for a service-oriented Bluetooth network. Although every device in a piconet can be the master according to the Bluetooth specification, in our network a central device is chosen to be always the master. Our network connects a large number of devices while guaranteeing service access. Therefore, devices send the maximum access intervals for their services (in the Bluetooth service descriptions) to the master when entering the network. The master collects the initial data (e.g. temperature for temperature service) from the services and stores it in a cache. Afterwards, the device is put into Park mode and only reactivated when exact service data is demanded or the service access interval is elapsed.

The remainder of this paper is organized as follows. Section 2 describes related research works. Needed background information is provided in section 3. Section 4 describes the Device Manager and its operations in detail. A conclusion is given and future research is specified in section 5.

## 2 Related Works

Low power approaches for wireless networks are addressed by numerous research works. On the hardware level, power consumption can be reduced by adjusting the power level on the wireless transmitter during active connections [2]. On the software level, the basic idea is to estimate when a device will transmit data and to suspend it for the time it is not used [3].

Bluetooth devices stay in five different modes or states, respectively, regarding low power approaches: Standby, Active, Sniff, Hold and Park mode. The *Standby mode* marks the unconnected state. The other modes are managed by the master. Active devices listen to all communication whereas the Sniff, Hold and Park modes are low power modes (suspend modes). Devices in *Sniff mode* frequently listen for a certain time quantum to the communication and are suspended otherwise. Devices in *Hold mode* are suspended for a certain time and automatically reactivated afterwards. Parked devices are not connected to the piconet but synchronize to the master in a certain interval. They have to be explicitly reactivated by the master.

Most of the research on intra-piconet communication focuses on optimizing the polling scheme depending on traffic estimations. They either utilize the Sniff mode or the Hold mode. We are not aware of any research which considers the Bluetooth Park mode for reducing power consumption.

Subsequent *polling schemes* utilize the Sniff mode of Bluetooth: Garg et al. proposed several polling schemes varying sniff interval and serving time in a sniff interval [4]. Chakraborty et al. proposed the Adaptive Probability based Polling Interval (APPI) [5]. APPI was developed for bursty traffic and adapts the serving time in a sniff interval to a probable and frequent burst of traffic. Yaiz et al. developed the polling scheme called Predictive Fair Polling (PFP) [6]. PFP uses a urgency metric for each slave predicting if data is available and keeping track of the fairness. The slave with the highest urgency metric is polled.

Following polling scheme make use of the Hold mode: Adaptive Power Conserving service discipline for Bluetooth (APCB), a polling algorithm proposed by Zhu et al. that utilizes the Hold mode [7]. Like the Adaptive Share Polling (ASP) from Perillo and Heinzelman [8]. APCB observes the traffic and estimates the traffic rates. The Hold interval is adapted accordingly. Adjusting power in APCB is done by altering a value which determines the change of the flow rate. In comparison to that, the range between the necessary amount and the actual amount of polls tunes the power consumption in ASP. The reason is the non-predictability of succeeding traffic while polling less or as necessary.

Another polling scheme which should be mentioned in this context is Deficit Round Robin (DRR), proposed by Shreedhar et al. [9]. It works similar to the simple Round Robin (RR) polling scheme, where all slaves are always polled by the master in a certain order and send all their data when polled. DRR limits the transmission of each node to a certain time quantum. If the transmission time of a node exceeds the corresponding quantum the transmission is stopped and the next node is served. The remaining transmission time is added to quantum for the next round. The BlueHoc software uses DRR as the scheduler [10].

The article of Lee at al. examines the affect of the amount of slaves on the throughput and latency time in a Bluetooth network [11]. That research also considers the Park mode. The Park mode is not used to reduce power consumption rather to extend the number of possible slaves and delivers results on throughput and latency time for all (parked and active) slaves. Slave are put to Park mode and reactivated in a RR manner.

We are not aware of any research work addressing the Park mode in low-power approaches although it can be additionally used for extending the number of connected slaves.

## 3 Background

### 3.1 Bluetooth

Bluetooth is radio based communication technology with a transmission range of 10 to 100 metres using the 2.4 GHz Industrial, Scientific and Medical (ISM) band. A spread spectrum is used to avoid interferences and noise of other devices

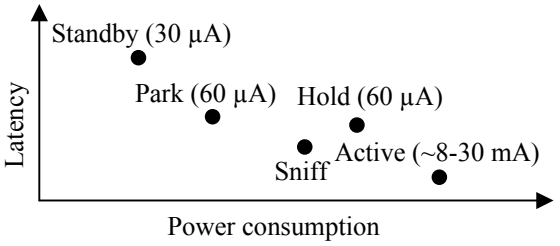


**Fig. 1.** Latency and power consumption of Bluetooth modes

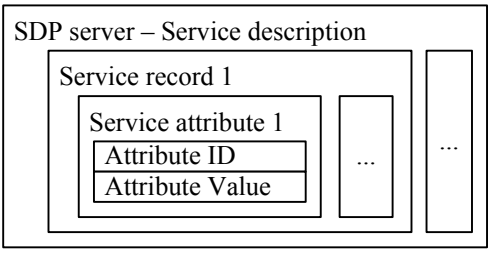


**Fig. 2.** Overview of the Bluetooth service descriptions

by frequency hopping (1600 hoppings per second). Signals are modulated using a Gaussian Frequency Shift Keying (GFSK) modulation scheme and utilizes slotted Time Division Duplex (TDD) with a slot interval of 625 $\mu$sec. The master manages the whole communication in a piconet. Slaves are only allowed to send data when polled by the master.

Figure 1 compares the Bluetooth low-power modes with the Active mode regarding latency and power consumption (adapted from Milios) [12]. The Park mode saves most power of the connection modes but has the longest latency time. The Hold mode uses more power than the Park mode in average because the device switches automatically to Active mode when the Hold interval is elapsed.

### 3.2 Service Orientation in Bluetooth

Bluetooth offers a minimal service-oriented functionality. The *Bluetooth Service Discovery Protocol* (SDP) offers searching and browsing for Bluetooth services based on service descriptions. Searching for service means that a SDP client (service user) queries available SDP servers (service providers) for desired services. Browsing for services is the searching without prior information about the services. A device can be both, SDP client and server. Only one SDP server per device is allowed. Bluetooth does neither provide any kind of notification mechanisms (e.g. when a SDP server enters or leaves the network, when a service description changes) nor methods to access the services.

A SDP server maintains a list of *service records* (as shown in Figure 2). Each service record represents a service and consists of a list of *service attributes*. Service attributes consist of an *attribute identifier* (ID) and corresponding *attribute value*. Attribute IDs are 16-bit unsigned integers and reflect the semantics of a service. Some attribute IDs and related value data types (e.g. Service Name as a string value) are predefined by the Bluetooth Special Interest Group (SIG). Each service instance belongs to a *service class* that specifies the meaning, prescribed services attributes and data types. New service classes are defined as a subclass of an existing one extended by new attributes. Service classes are represented by a 128-bit Universally Unique ID (UUID). UUID guarantee to be unique across all space and all time.

## 4 Device Manager for Bluetooth

This paper addresses a single centralized piconet consisting of an always available device (always functioning as Bluetooth master) and several resource-constraint devices (Bluetooth slaves) which may enter and leave the network dynamically. We build on the assumptions that Bluetooth slaves work as SDP servers only. They can not be SDP clients due to there limitations. The Bluetooth master primarily works as a SDP client.

The Bluetooth master contains the *Device Manager* (DM). The DM is permanently available and controls the entire network, all connections and is responsible for reducing the power consumption of the slaves utilizing the Park mode. It involves following tasks: providing scheduling for connected devices, establishing a connection, accessing services, reacting on unnotified device leaving and changes of the service descriptions as well as refusing a device. The task are described in subsequent sub-sections.

### 4.1 Overview of Device Manager

The Device Manager offers access to available Bluetooth services and hides the actual Bluetooth devices. It accepts service requests and processes them by accessing the Bluetooth services. The DM manages an additional cache which works as a service directory for available services and their states.

We distinguish between three devices regarding Park mode: Cached, on-demand and always-active devices. *Cached devices* update their attribute values in a certain interval. The master requests the values and caches them. *On-demand devices* are woken up by the master when new attributes values are requested. *Always-active devices* deliver accurate values and can not be parked as the name suggests. The type of the device is defined by the semantics of their services. In case that embedded services require different type following rules apply: If a service is always-active the device is always active. Cached and on-demand services are accessed according to their requirements and work parallel.

In case that there are more active devices needed than supported by piconets, the scheduler of Lee et al. [11] can be applied.

### 4.2 Scheduling

Time controlled operations in conjunction with time constraints belong to classical scheduler problems that are solved by scheduler algorithms normally. Such algorithm is not part of this paper. The DM's scheduler is requested at time of establishing a connection to a new device with required park interval to assure it will not disorganize other devices with required park intervals. Furthermore, the scheduler is responsible for initiating service access to services with park interval attributes.

### 4.3 Service Description for Connected Devices

The service management of the Device Manager requires two additional attributes which have to be defined by the service provider: MaximumParkInterval and AlwaysActive.

The *MaximumParkInterval* attribute (Unsigned Integer) determines the maximum time a service may be parked (in ms). It is used for cached services to define the maximum interval in which they have to be updated. The Park interval for the device results from the minimal MaximumParkInterval of all services (for devices providing more than one service). The attribute value 0 stands for on-demand services.

The *AlwaysActive* attribute (Boolean) identifies if a device may be parked. If the value is 0 the device may be parked. The value 1 stands for always active devices. Devices are always active devices by default.

### 4.4 Establishing a Connection

Before an SDP connection can be established the new device has to build up a Bluetooth connection to the master. This may initiated in two ways: The master searches for new devices or the new device runs an inquiry and finds the master.

When the master finds a device it will be automatically connected as an active slave. When a new device starts *paging* (establishing a piconet) it normally functions as a master of the connection (according to the Bluetooth specification). Our described network has a predefined master. Therefore, the Bluetooth role switch procedure [1] has to be applied that the new device becomes a slave.

When the Bluetooth connection is established the master sends an SDP request to the new slave browsing for services. The SDP response of the new slave includes the service descriptions which are put into the service cache of the master. If the slave is an on-demand or cached device it is put into Park mode.

### 4.5 Accessing a Service

Time of accessing a service by the master depends on service management attributes of the service. Services with a specified value for park interval are woken up by the scheduler accordingly. The new state values are determined by regular Bluetooth SDP operations (request and response) and are stored in the master's cache until they are accessed from outside the master. On-demand services are

accessed only if service access is requested from outside the master. Therefore the master asked the scheduler to get an appropriate moment and puts the device into active mode. After using the service the master puts the device back into Park mode. Services requiring always active connections are already in active mode. Service access can happen permanently.

### 4.6 Disconnection of a Device

Since communication between master and slave is always initiated by the master, there is no way for slaves to inform the master about intended leaving the network. Disconnected devices are recognized at next service access as described above. After realizing such breakdown, the master informs the scheduler and deletes its cache for each service the device was offering.

### 4.7 Refusing a Device

In certain situations the master has to refuse a device if it tries to establish a new connection. The decision whether to refuse or not is made by the scheduler in dependence of the quality of service constraints given by the device (e.g. asking for large bandwidth). In all cases the master will send a Disconnection Command to the device.

### 4.8 Changing Service Description While Connected

Changing a service description means adding or removing one or more service attributes in the service record. As said before, communication is initiated by the master only. Therefore changed descriptions are recognized at next service access. For each added attribute the master allocates new fields in the cache, for each missing attribute the master removes according fields, respectively.

## 5 Conclusion and Future Work

We have described how the Bluetooth Park mode can be utilized to connect a large number of resource-constraint devices while reducing power consumption in a service-oriented Bluetooth network. The whole communication is managed by a predefined master device which contains a Device Manager for the management including a cache for available service descriptions and related service values. The Device Manager can be used to access Bluetooth services from outside the Bluetooth network. This network set-up addresses Bluetooth slaves which only offer services and do not make use of other services. This paper showed the concept of such a network, described the needed algorithms and procedures and illustrated the feasibility of our approach. Currently, we are implementing the concept which will be used for in-car networks.

Future work will be done on evaluations of the implementation and improvements concerning larger bandwidth and shorter latency times. Furthermore, future proposals will adapt this concept to Bluetooth scatternets.

## Acknowledgement

This work was supported by the SIRENA project in the framework of the European premier cooperative R&D program "ITEA".

## References

1. The Bluetooth Special Interest Group: Specification of the Bluetooth System 1.2 (2004)
2. Monks, J., Bharghavan, V., Hwu, W.-M.: A Power Controlled Multiple Access Protocol for Wireless Packet Networks. In Proceedings of IEEE INFOCOM 2001, Anchorage, Alaska, USA (2001)
3. Kravets, R., Krishnan, P.: Power Management Techniques for Mobile Communication. In Proceedings of MobiCOM, Dallas, Texas, USA 1998 (1998)
4. Garg, S., Kalia, M., Shorey, R.: MAC Scheduling Policies for Power Optimization in Bluetooth: A Master Driven TDD Wireless System. In Proceedings of IEEE Vehicular Technology Conference 2000, Tokyo, Japan (2000)
5. Chakraborty, I., Kashyasp, A., Kumar, A., Rastogi, A., Saran, H., Shorey, R.: MAC Scheduling Policies with Reduced Power Consumption and Bounded Packet Delays for Central Controlled TDD Wireless Networks. In Proceedings of IEEE International Conference on Communications 2001, Helsinki, Finland (2001)
6. Yaiz, R., Heijenk, G.: Polling Best Effort Traffic in Bluetooth. In Proceedings of The Fourth International Symposium on Wireless Personal Multimedia Communications 2001, Aalborg, Denmark (2001)
7. Zhu, H., Cao, G., Kesidis, G., Das, C.: An Adaptive Power-Conserving Service Discipline for Bluetooth. In Proceedings of IEEE International Conference on Communications 2002, New York, USA (2002)
8. Perillo, M., Heinzelman, W.: ASP: An Adaptive Energy-Efficient Polling Algorithm for Bluetooth Piconets. In Proceedings of IEEE Hawaii International Conference on System Sciences 2003, Big Island, Hawaii, USA (2003)
9. Shreedhar, M., Varghese, G.: Efficient Fair Queueing using Deficit Round Robin. In Proceedings of ACM SIGCOMM 1995 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, Cambridge, Massachusetts, USA (1995)
10. BlueHoc: Bluetooth performance evaluation tool. https://oss.software.ibm.com/developerworks/opensource/bluehoc/ (2002)
11. Lee, T.-J., Jang, K., Kang, H., Park, J.: Model and Performance Evaluation of a Piconet for Point-to-Multipoint Communications in Bluetooth. In Proceedings of IEEE Vehicular Technology Conference 2001, Rhodes, Greece (2001)
12. Milios, J.: Baseband Methods for Power Saving. Presentation at Bluetooth Developers Conference, San Jose, California, USA (2000)